PATENT ABSTRACTS OF JAPAN

(11)Publication number : **09-046235**
(43)Date of publication of application : **14.02.1997**

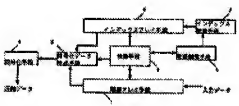
(51)Int.Cl. H03M 7/40

(21)Application number : **07-249956** (71)Applicant : **NEC CORP**
(22)Date of filing : **27.09.1995** (72)Inventor : **OKAMURA TOSHIHIKO**

(30)Priority
Priority number : **06239926** Priority date : **04.10.1994** Priority country : **JP**
07121135 **19.05.1995** **JP**

(54) **DATA COMPRESSION DEVICE**

(57)Abstract:
PROBLEM TO BE SOLVED: To improve encoding speed by providing a retrieval means for retrieving the character string of a record array means and a registration control means controlling the registration of the character string in the record array means to the retrieval means.
SOLUTION: Character strings x(s), x(s+1)< are compared with the character string registered in the retrieval means 2 by using the retrieval means 2, and the longest matched string is obtained. An index supplied from an index update means 7 is distributed to an entry where x(s) is stored, and it is held in an index array means 6. The means 7 updates an index value whenever the index is supplied to the means 6. The means 2 transmits a matched length to an encoding data generation means 3, and transmits a difference between the held index and the index held in the entry corresponding to the position of the index x(s) to an encoding means 4. The means 4 finally converts it into a binary string and updates a record array means 1 when encoding is terminated. A registration control means 5 allocates a number from the means 7 and holds it in the means 6.



(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

特開平9－46235

(43)公開日　平成９年(1997)２月14日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｈ０３Ｍ　7/40		9382－5K	Ｈ０３Ｍ　7/40	

審査請求　有　　請求項の数11　ＯＬ　（全 18 頁）

(21)出願番号　特願平7－249956

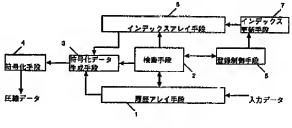
(22)出願日　平成７年(1995)９月27日

(31)優先権主張番号　特願平6－239926
(32)優先日　平６(1994)10月４日
(33)優先権主張国　日本（ＪＰ）
(31)優先権主張番号　特願平7－121135
(32)優先日　平７(1995)５月19日
(33)優先権主張国　日本（ＪＰ）

(71)出願人　000004237
日本電気株式会社
東京都港区芝五丁目７番１号
(72)発明者　岡村　利彦
東京都港区芝五丁目７番１号　日本電気株式会社内
(74)代理人　弁理士　京本　直樹　（外２名）

(54)【発明の名称】　データ圧縮装置

(57)【要約】
【課題】　大きな履歴アレイを用いても、比較すべき文字列の個数、検索手段へ登録する文字列の数が効果的に削減でき、圧縮率の劣化を小さく抑えて符号化速度の向上を図ることができるデータ圧縮方式を提供する。
【解決手段】　入力データを格納する複数のエントリを有する履歴アレイ手段１と、履歴アレイ手段１中の文字列の検索を容易にするための検索手段２と、履歴アレイ手段１の文字列の検索手段２への登録を以下に述べる方法によって制御する登録制御手段と５と、検索手段２に登録されている文字列の先頭位置に対応するエントリにのみインデックスを保持するインデックスアレイ手段６と、インデックスアレイ手段６にインデックスを供給するインデックス更新手段７と、符号化データ生成手段３と、符号化手段４とから構成されている。



1

【特許請求の範囲】

【請求項１】入力データを格納する複数のエントリを有する履歴アレイ手段を持ち、これから圧縮を行う入力データと前記履歴アレイ手段に格納されている既に符号化を終えている入力データを比較し、十分に長い一致文字列が発見された場合には該一致文字列を前記履歴アレイ手段における該一致文字列の位置、長さの情報で表すことによって圧縮を達成するデータ圧縮装置において、

前記履歴アレイ手段の文字列の検索を行うための検索手段と、

前記履歴アレイ手段内の文字列の前記検索手段への登録を制御する登録制御手段と、を有することを特徴とするデータ圧縮装置。

【請求項２】各エントリが前記履歴アレイ手段の各エントリに対応するインデックスアレイ手段を持ち、前記履歴アレイ手段のエントリの内、前記検索手段に登録される文字列の先頭のエントリにのみインデックスを振り、前記インデックスアレイ手段は前記履歴アレイ手段の該エントリに対応する位置に該インデックスを保持し、前記アレイ手段中の位置を表す情報を該インデックスから生成される位置コードによって表すことを特徴とする請求項１記載のデータ圧縮装置。

【請求項３】前記登録制御手段は該一致文字列の長さに基づいて前記検索手段に登録される新たな文字列を決定することを特徴とする請求項１または請求項２記載のデータ圧縮装置。

【請求項４】前記検索手段に新たに登録される文字列は入力データ中の該一致文字列の先頭から始まる文字列と、該一致文字列の長さが予め決められた閾値以下の場合に該一致文字列の途中の位置から始まる文字列であることを特徴とする請求項３記載のデータ圧縮装置。

【請求項５】前記閾値を動的に変更することを特徴とする請求項４記載のデータ圧縮装置。

【請求項６】前記登録制御手段は該一致文字列の先頭の前記履歴アレイ手段における位置に基づいて前記検索手段に登録される新たな文字列を決定することを特徴とする請求項１または請求項２記載のデータ圧縮装置。

【請求項７】前記検索手段に新たに登録される文字列は入力データ中の該一致文字列の先頭から始まる文字列と、該一致文字列の先頭の前記履歴アレイ手段内の位置と入力データ中の該一致文字列の先頭位置との相対距離が予め決められた閾値以上の場合には該一致文字列の途中の位置から始まる文字列であることを特徴とする請求項６記載のデータ圧縮装置。

【請求項８】前記閾値を動的に変更することを特徴とする請求項７記載のデータ圧縮装置。

【請求項９】入力データを格納する複数のエントリを有する履歴アレイ手段を持ち、これから圧縮を行う入力データと前記履歴アレイ手段に格納されている既に符号化を終えている入力データを比較し、十分に長い一致文字

2

列が発見された場合には該一致文字列を前記履歴アレイ手段における該一致文字列の位置、長さの情報で表すことによって圧縮を達成するデータ圧縮装置において、

前記履歴アレイ手段の文字列の検索を行うための検索手段と、

入力データ中の該一致文字列の先頭から高々閾値個までの位置から始まる文字列を前記検索手段へ登録する登録制御手段と、を有することを特徴とするデータ圧縮装置。

【請求項１０】各エントリが前記履歴アレイ手段の各エントリに対応するインデックスアレイ手段を持ち、前記履歴アレイ手段のエントリの内、前記検索手段に登録される文字列の先頭のエントリにのみインデックスを振り、

前記インデックスアレイ手段は前記履歴アレイ手段の該エントリに対応する位置に該インデックスを保持し、前記アレイ手段中の位置を表す情報を該インデックスから生成される位置コードによって表すことを特徴とする請求項９記載のデータ圧縮装置。

【請求項１１】各エントリが前記履歴アレイ手段の各エントリに対応するインデックスアレイ手段を持ち、前記検索手段に登録されている文字列の先頭のエントリの内、連続する該エントリから成るブロックにインデックスを割り振り、

前記インデックスアレイ手段は前記履歴アレイ手段の該エントリに対応する位置に、該エントリが属する該ブロックのインデックスを保持し、前記アレイ手段の位置を表す情報を該エントリが属する該ブロックのインデックスと該エントリの該ブロック内での位置を表す情報から生成される位置コードによって表すことを特徴とする請求項１または請求項９記載のデータ圧縮装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は無歪みで復元可能なデータ圧縮復元方式に関する。

【０００２】

【従来の技術】レンペル－ジブ方式として知られるテキスト置換型の無歪みデータ圧縮方式は１９８０年代後半から広く使用されるようになってきた。中でもＬＺ７７型と呼ばれるデータ圧縮アルゴリズムは圧縮率の良さから計算機上のファイル圧縮ツールにしばしば組み込まれている。ＬＺ７７型は既に符号化を終えた入力データを格納するための履歴アレイを持つことを特徴とする。履歴アレイは更新が容易になるように円環的に使用されることが多い。つまり履歴アレイ内に一定長の先読み領域と呼ばれる領域を用意し、そこにはこれから圧縮を行うデータを格納しておき、先読み領域は処理が進むにつれて円環的に移動していく。

【０００３】これから圧縮を行うデータ（先読み領域内のデータ）は履歴アレイ中の先読み領域以外の部分から

始まる文字列、つまり既に圧縮が終了している入力データを格納するエントリから始まる文字列と照合し、一致列を求める。十分に長い一致列が見つかったら、入力データの一致列の部分に対する符号語として、履歴アレイ内のその一致列の先頭位置、長さを符号語として送出する。必要に応じてこれらの値は可変長符号化してさらに効率を良くする。一致列の先頭位置はなるべく最新のものに統一し、先読み領域の先頭との相対距離で表すことによって、比較的小さい値に集中させることができ、可変長符号化が有効に働く。履歴アレイ内は常に最新の符号化済みのデータが格納されるように更新していく。つまり、ＬＺ７７型は常に最新の符号化済みのデータを“辞書”としており、辞書に登録されている文字列は履歴アレイ内の既に圧縮を終えたデータを格納するエントリから始まる一定長以下の文字列すべてとなる。これはレンペル－ジブ方式のもう一つの形態であるＬＺ７８型の圧縮方法に比べて辞書に登録されている文字列が多いことになり、より長い一致列が見つかり良い圧縮率を示すことが多い。

【０００４】十分長い一致列が見つからなかった場合の処置法としてＬＺＳＳと呼ばれるアルゴリズムがある。ｌｈａ，ｇｚｉｐなどのファイル圧縮用フリーソフトはＬＺＳＳを基本としている。ＬＺＳＳは二つの符号化モードを用いる。十分に長い一致列（１シンボル１ｂｙｔｅの場合で３シンボル以上が普通）が見つかった場合に用いるコピーモードと、見つからなかった場合に用いるリテラルモードである。コピーモードの場合は一致長を表すビット列と一致位置を表すビット列が符号語となる。リテラルモードの場合には先頭一文字がそのまま送出される。リテラルモードとコピーモードを区別するためにｌビットのフラグをつけるか、または一致長と入力データのアルファベットを合わせて新たに大きなアルファベットを生成するといった処置がとられる。

【０００５】ＬＺ７７型を実装する上では、これから圧縮を行う入力文字列と履歴アレイ内の文字列との照合処理を高速に行うことが問題となる。図１０に示すように、観念的には履歴アレイ手段１に対して何らかの検索手段２を設定し、履歴アレイ内の文字列はすべての検索手段２に登録する形になる。なお、図１０において、３は符号化データ生成手段、４は符号化手段である。

【０００６】現在は専用ハードウェアで並列的に履歴アレイ内を検索する他に、履歴アレイ内が順次更新されることから、二分木、トライなどの木構造を使う検索手段か、ハッシュ表に衝突時用の線形リストを備えた検索手段がよく用いられる。検索手段は文字列の先頭の履歴アレイ内の位置を示すポインタを用いて構成される。検索手段は文字列の先頭位置を示すポインタから履歴アレイ内の文字列を読み込みターゲットの文字列と比較する。

【０００７】二分木を用いた方法はＩＥＥＥの刊行物「ＩＥＥＥ　トランザクション　オンコミュニケーション」１９８６年ｖｏｌ．３４，ｎｏ．１２，ｐｐ．１１７６－１１８２に開示されている。トライを用いた方法は米国特許第４，９０６，９９１号明細書に詳しく述べられている。ハッシュ表＋線形リストの方法は特開平３－６８２１９号公報に詳しく述べられている。

【０００８】それぞれの検索法を図１１～図１４に示す。図１１は履歴アレイ手段の状態を表し、図１２，図１３，図１４は履歴アレイ手段が図１１の状態のときの各検索手段の構成を表す。図１２は二分木を用いた方法、図１３はトライを用いた方法、図１４はハッシュテーブルに衝突用の線形リストを備えた方法である。アレイから溢れた文字列は検索手段から削除する必要があるため、検索手法は削除のコストが低いものでなければならない。

【０００９】

【発明が解決しようとする課題】ＬＺ７７型では、長い一致列が見つかれば圧縮率は向上する。よりたくさんの文字列とマッチングをとればより長い一致列が見つかる可能性が高くなり、そのためには履歴アレイの大きさＮ，コピーの最大長Ｌを大きくすればよい。しかし、Ｎ，Ｌを大きくすると三つの問題が生じる。

【００１０】第一にＮ，Ｌの増大は文字列の位置、長さを表すために必要なビット数の増加につながり、かえって圧縮率を劣化させる恐れがある。しかし、この問題は位置、長さなどを可変長符号化することである程度解決できる。１シンボルを１バイト（８ビット）とした時、位置，長さを固定長符号化するという前提の下では履歴アレイ長Ｎ＝８１９２程度、一致列の最大長Ｌ＝６４程度が最良と言われていたが、現在ではＮ＝３２７６８，６５５３６、Ｌ＝２５６～１０２４程度が最もよい圧縮率を示すことが判明している。

【００１１】第二にＮ，Ｌの増大は符号化時にメモリを大きく消費することにつながる。しかし、Ｎ＝６５５３６，Ｌ＝２０４８程度であったら検索手段まで含めて高々２メガバイトのメモリで済むため、現在ではワークステーションや高性能なパソコンで実行する上では問題なく実行できる。

【００１２】第三の問題が最も重大で、Ｎ，Ｌの増大は符号化速度を著しく劣化させる。ＬＺ７７型の符号化速度を見積もる指標として文字比較回数がある。データ長をＭとすると、二分木、トライ、ハッシュテーブル＋線形リストの文字比較回数は次のように近似される。

【００１３】

・特別に検索手段を設けない　…　Ｍ′ＬＮ

・二分木　…　ＭＬｏｇ$_2$Ｎ

・トライ　…　ＭＬ

・ハッシュ表　…　Ｍ′ＬＮ′

【００１４】ＬＺ７７型では入力データは部分列に分解され、部分列毎に符号語が対応することになるが、Ｍ′はこの部分列の個数である。またＮ′は線形リストの平

均の長さである。二分木やトライを用いた場合、文字列を検索手段に登録するためには検索手段を用いて検索を行い、しかるべき位置に挿入しなければならない。そのために、データ中の全ての位置から一定長以下の文字列を検索手段を用いて検索しなければならない。これに対してハッシュテーブルを使用する方法では文字列検索を必要とするのは、符号化を行う文字列の先頭のみからに限られる。一致列の途中から始まる文字列の検索手段への登録に関してはハッシュ関数の計算とハッシュ表のＩレコードの書き換えだけで済み、文字列検索を必要としない。そこでハッシュテーブルを用いた方法では文字列比較回数はＭではなく、Ｍ′の倍数という形で表せるのである。Ｎ′は最悪Ｎに等しくなるが、十分なハッシュテーブルを用意すれば平均は十分に小さく、また、圧縮率の多少の劣化を許容すればＮ′は適当な回数で打ち切ることができるので小さい定数で抑えることもできる。そのためにＭ′と二分木、トライの場合のＭの差が効いて、ハッシュテーブルを使用した方法の高速性が生まれる。とくに冗長の大きなデータの場合にはＭとＭ′の差が大きく、ハッシュテーブルを使用した場合と木構造を用いた場合とでは実行速度の大きな違いが生じる。Ｎ＝６５５３６，Ｌ＝１０２４に設定すると二分木、トライでは著しく符号化速度が劣化してしまい、ハッシュテーブルを使用した方法の独壇場となる。しかし、ハッシュテーブルを使用した方法で最長一致列を求めようとするとＮ′が大きくなり、１回の検索にかかるコストが余りにも大きく、特別に検索手段を設けない場合と変わらなくなり、実行速度は極めて劣化してしまう。

【００１５】検索手段に木構造を用いた場合の高速化手法の一つが米国特許４，９０６，９９１号の拡張具体例に開示されている。そこで述べられている方法は、検索手段にトライ構造を用い、符号化する文字列の先頭から始まる文字列のみを検索手段に登録する方法である。また、一致長が２または３の場合にその途中から始まる文字列を検索手段に登録する方法についても簡単に述べられている。

【００１６】本発明の目的は、大きな履歴アレイを用いても、比較すべき文字列の個数、検索手段へ登録する文字列の数が効果的に削減でき、圧縮率の劣化を小さく抑えて符号化速度の向上を図ることができるデータ圧縮方式を提供することにある。本発明は先に述べた米国特許４，９０６，９９１の方法を拡張した、より柔軟性のある登録制御を行うことによって目的を達成する。

【００１７】

【課題を解決するための手段】第１の発明は、入力データを格納する複数のエントリを有する履歴アレイ手段を有し、これから圧縮を行う入力データと、前記履歴アレイ手段に格納されている文字列を比較し、十分に長い一致文字列が発見された場合には該一致文字列を前記履歴アレイ手段における該一致文字列の位置、長さの情報で

表すことによって圧縮を達成するデータ圧縮装置において、前記履歴アレイ手段の文字列の検索を行うための検索手段と、前記履歴アレイ手段内の文字列の前記検索手段への登録を制御する登録制御手段と、を有することを特徴とする。

【００１８】第２の発明は、第１の発明のデータ圧縮装置において、各エントリが前記履歴アレイ手段の各エントリに対応するインデックスアレイ手段を持ち、前記履歴アレイ手段のエントリの内、前記検索手段に登録される文字列の先頭のエントリにのみインデックスを張り、前記インデックスアレイ手段は前記履歴アレイ手段の該エントリに対応する位置に該インデックスを保持し、前記アレイ手段中の位置を表す情報を該インデックスから生成される位置コードによって表すことを特徴とする。

【００１９】第３の発明は、第１または第２の発明において、前記登録制御手段は該一致文字列の長さに基づいて前記検索手段に登録される新たな文字列を決定することを特徴とする。

【００２０】第４の発明は、第３の発明において、前記検索手段に新たに登録される文字列は入力データ中の該一致文字列の先頭から始まる文字列と、該一致文字列の長さが予め決められた閾値以下の場合に該一致文字列の途中の位置から始まる文字列であることを特徴とする。

【００２１】第５の発明は、第４の発明において、該閾値を動的に変更することを特徴とする。

【００２２】第６の発明は、第１または第２の発明において前記登録制御手段は該一致文字列の先頭の前記履歴アレイ手段における位置に基づいて前記検索手段に登録される新たな文字列を決定することを特徴とする。

【００２３】第７の発明は、第６の発明において前記検索手段に新たに登録される文字列は入力データ中の該一致文字列の先頭から始まる文字列と、該一致文字列の先頭の前記履歴アレイ手段内の位置と入力データ中の該一致文字列の先頭位置との相対距離が予め決められた閾値以上の場合には該一致文字列の途中の位置から始まる文字列であることを特徴とする。

【００２４】第８の発明は、第７の発明において、該閾値を動的に変更することを特徴とする。

【００２５】第９の発明は、入力データを格納する複数のエントリを有する履歴アレイ手段を持ち、これから圧縮を行う入力データと前記履歴アレイ手段に格納されている既に符号化を終えている入力データを比較し、十分に長い一致文字列が発見された場合には該一致文字列を前記履歴アレイ手段における該一致文字列の位置、長さの情報で表すことによって圧縮を達成するデータ圧縮装置において、前記履歴アレイ手段の文字列の検索を行うための検索手段と、入力データ中の該一致文字列の先頭から高々閾値個までの位置から始まる文字列を前記検索手段へ登録する登録制御手段と、を有することを特徴とする。

【００２６】第１０の発明は、第９の発明のデータ圧縮方法において、各エントリが前記履歴アレイ手段の各エントリに対応するインデックスアレイ手段を持ち、前記履歴アレイ手段のエントリの内、前記検索手段に登録される文字列の先頭のエントリにのみインデックスを振り、前記インデックスアレイ手段は前記履歴アレイ手段の該エントリに対応する位置に該インデックスを保持し、前記アレイ手段中の位置を表す情報を該インデックスから生成される位置コードによって表すことを特徴とする。

【００２７】第１１の発明は、第１または第９の発明において、各エントリが前記履歴アレイ手段の各エントリに対応するインデックスアレイ手段を持ち、前記検索手段に登録されている文字列の先頭のエントリの内、連続する該エントリから成るブロックにインデックスを割り振り、前記インデックスアレイ手段は前記履歴アレイ手段の該エントリに対応する位置に、該エントリが属する該ブロックのインデックスを保持し、前記アレイ手段の位置を表す情報を該エントリが属する該ブロックのインデックスと該エントリの該ブロック内での位置を表す情報から生成される位置コードによって表すことを特徴とする。

【００２８】

【作用】本発明では検索手段への履歴アレイ内の文字列の登録を制御するという形で、比較する文字列の個数と検索手段へ登録する文字列の個数を削減させている。圧縮率を良くするための大きな履歴アレイでは同じ文字列が頻繁に現れることが多く、符号化速度の面から大きな負荷を与えるが、本発明はこれを解消するための手段を与える。

【００２９】圧縮を行う文字列は履歴アレイ内の文字列とのマッチングをとって一致列を求める。このように符号化する文字列の場合、検索は必須で、その結果を利用して検索手段に容易に挿入することができるので、符号化の先頭位置から始まる文字列については検索手段に登録するのが妥当である。一致列が見つかった場合、符号化された文字列の途中の位置から始まる文字列を処理する方法が本発明の着眼点である。登録制御手段の判断の基準は、復元側でもわかる仕組みでなければならない。そこで、一致長や一致列の位置といった情報から生成される基準を使用する方法が考えられる。

【００３０】一致長が長い文字列が符号化された場合、その途中から始まる文字列に関しても既に履歴アレイ内に存在しているものが多いことになる。そこで途中から始まる文字列の登録を控えても圧縮率に大きく影響しないことが期待できる。本発明の第３から第５の発明はこのことを利用した方式である。同じような文字列が繰り返し現れることが冗長度の高いデータに対しては、検索構造への無駄な登録を大きく省くことができ、履歴アレイ内から始まる文字列をすべて登録していた従来の方式

に比べて符号化速度を向上させることができる。

【００３１】一致文字列の履歴アレイ内での位置に基づいて符号化された文字列の途中の位置から始まる文字列を登録するかを判断するという手法は次のようになる。

一致列が昔に現れたものであったら、その列は近い内に履歴アレイから溢れてしまう。そのような状況になってもその文字列が絡んだ文字列が参照できるようにするために、途中から始まる文字列を登録するのである。逆に比較的近い位置に一致列が同定された場合、その符号化文字列が絡んだ文字列はしばらくは履歴アレイ内に存在するので、あえて符号化された文字列の途中から始まる文字列まで検索手段に登録しなくても圧縮率はそれ程劣化しないことが期待できる。冗長なデータでは比較的近い位置に最長一致列が見つかることが多いため、検索手段に登録される文字列を絞ることができ、従来の方式に比べて符号化速度を向上させることができる。

【００３２】また、単純に一致文字列の先頭から閾値個までの位置から始まる文字列を登録する方法が考えられる。ただし、閾値が一致文字列より大きい場合は一致文字列のすべての位置から始まる文字列を登録する。そういった意味ではこの方法も、一致長に基づいて登録される文字列が決めている、とみなすこともできる。この場合、閾値を小さく設定すると登録される文字列が少なくなり、符号化速度の向上を生む。

【００３３】本発明の方式では、履歴アレイ中のすべての位置から始まる文字列を参照することができるわけではないので、一致列の先頭位置をアレイ中の位置でそのまま表すと冗長になる。必要な位置にだけインデックスを振り、一致列の先頭位置のインデックスと最新のインデックスの差を用いて位置を表すことによって、履歴アレイの位置で見た相対距離で表すより小さな値で表現でき、位置を表す符号語のビット長に関しては有利になる。第２の発明のように、インデックスアレイを用いて、各エントリに対応するインデックスを保持する必要がある。また、検索手段に登録されている文字列の先頭位置の中で連続するものをまとめたブロックを形成し、ブロックに割り振ったインデックスを用いて一致列の先頭位置を表すこともできる。

【００３４】

【発明の実施の形態】次に、本発明の実施例について図面を参照して説明する。

【００３５】図１は、一実施例のデータ圧縮装置を示す構成図である。この装置は、入力データを格納する複数のエントリを有する履歴アレイ手段１と、履歴アレイ手段１中の文字列の検索を容易にするための検索手段２と、履歴アレイ手段１の文字列の検索手段２への登録を以下に述べる方法によって制御する登録制御手段５と、検索手段２に登録されている文字列の先頭位置に対応するエントリにのみインデックスを保持するインデックスアレイ手段６と、インデックスアレイ手段６にインデッ

クスを供給するインデックス更新手段７と、符号化データ生成手段３と、符号化手段４とから構成されている。

【００３６】次に、このデータ圧縮方式の動作を図２のフローチャートを参照しながら説明する。

【００３７】入力データをＸ＝ｘ（０）ｘ（１）ｘ（２）…とする。履歴アレイ手段１はＮシンボルの入力データを格納するとする。１シンボル１バイト（８ビット）として、Ｎ＝３２７６８または６５５３６が適当である。Ｎは任意の値でよいが円環的に用いるため、２の巾で探ることがよく行われる。また、一致長の最大長Ｌは２５６以上にとる。履歴アレイ手段における先読み領域はＬ以上の大きさにする。この実施例では先読み領域の大きさをＬとする。

【００３８】まず、履歴アレイ手段１，検索手段２，インデックスアレイ手段６，インデックス更新手段７を初期化する（ステップＳ１）。

【００３９】履歴アレイ手段の初期状態は空でも適当なデフォルトの文字列でも、要は復元時に同じ状態が再現できるものであったら何でもよい。検索手段２，インデックスアレイ手段６，インデックス更新手段７の初期状態はアレイ手段の初期状態に合わせて決まる。

【００４０】今、…ｘ（ｓ－２）ｘ（ｓ－１）までの符号化を終えてｘ（ｓ）ｘ（ｓ＋１）…という文字列の符号化を行うものとする（ステップＳ２）。このとき履歴アレイ手段内の各エントリにはｘ（ｓ－Ｎ＋Ｌ），…，ｘ（ｓ－１），ｘ（ｓ），…，ｘ（ｓ＋Ｌ－１）という文字列が格納されており、特に先読み領域にはｘ（ｓ），…，ｘ（ｓ＋Ｌ－１）という文字列が格納されている。検索手段２に登録されている文字列はｘ（ｓ－Ｎ＋１），…，ｘ（ｓ－１）の内、限定された位置から始まる、長さＬ以下の文字列である。検索手段２に登録されている文字列の先頭位置に対応するインデックスアレイ手段６のエントリには対応するインデックスが保持されている。

【００４１】検索手段２を用いて文字列ｘ（ｓ）ｘ（ｓ＋１）…と検索手段２に登録されている文字列と比較し（ステップＳ３），最長一致列を求める。場合によっては最長一致列ではなくなるべく長い一致列でよい。ｘ（ｓ）ｘ（ｓ＋１）…は検索結果が分かっているので検索構造へ登録する（ステップＳ４）。ｘ（ｓ）が格納されているエントリにはインデックス更新手段７から供給されるインデックスを割り振り、インデックスアレイ手段６に保持する。インデックス更新手段７はインデックスアレイ手段６にインデックスを供給する度にインデックス値を１更新する。インデックスは各エントリを区別できればよいので、Ｎに対する剰余で表すことにより０からＮ－１で表すことができる。

【００４２】最長一致列がｘ（ｓ－Ｎ＋Ｌ＋ｋ）…ｘ（ｓ－Ｎ＋Ｌ＋ｋ＋Ｉ－１）＝ｘ（ｓ）…ｘ（ｓ＋Ｉ－１）であったとする（ステップＳ５）。検索手段２は、一致長Ｉを符号化データ生成手段３に送る。Ｉが予め決められた一定値Ｍ以上であったら、符号化データ生成手段３は一致列が見つかったというフラグとともにｘ（ｓ）…ｘ（ｓ＋Ｉ－１）に対応する符号語としてＩとｘ（ｓ－Ｎ＋Ｌ＋ｋ）の位置に対応するエントリに保持されているインデックスとｘ（ｓ）の位置に対応するエントリに保持されているインデックスとの差（位置コード）を符号化手段４に送出する（ステップＳ６、これをコピーモードによる符号化と呼ぶ）。ＩがＭよりも小さかったらＩ＝１として、符号化データ生成手段は一致列が見つからなかったというフラグとともにｘ（ｓ）をそのまま符号化手段に送出する（ステップＳ７、これをリテラルモードによる符号化と呼ぶ）。

【００４３】符号化手段４はＩ，ｄ，ｘ（ｓ）などを最終的な二進列に変換する（ステップＳ６，Ｓ７）。符号化手段４には固定長のものの他、ハフマン符号化、算術符号化などを利用できる。符号化手段４はデータが入力される度に符号化を行う他、バッファに値を一時的に蓄え、ブロック単位に処理することもできる。

【００４４】ｘ（ｓ）…ｘ（ｓ＋Ｉ－１）の符号化が終わった時点で、履歴アレイ手段１を更新する。新たな入力データを読み込み、先読み領域はｘ（ｔ＋１）のエントリから始まる領域に移動する。このとき、履歴アレイ手段１から削除された古い文字列は検索手段２から削除される（ステップＳ８）。

【００４５】登録制御手段５はｘ（ｓ＋１），…，ｘ（ｓ＋Ｉ－１）から始まる文字列のうち、検索手段２に登録するものがあるかどうかを判断する。例えばリテラルモードで符号化された文字列の場合は登録するものがないと判断して、次の文字列の符号化に移る（ステップＳ９）。登録すると選択された文字列は検索手段２へ登録し、それぞれの先頭位置のエントリに順にインデックス更新手段７から供給される番号を振り、インデックスアレイ手段６に保持する（ステップＳ１０）。

【００４６】以上で入力データの部分列ｘ（ｓ）…ｘ（ｓ＋Ｉ－１）の符号化は終了で、履歴アレイ手段１に新しいシンボルを挿入し、ｘ（ｓ＋１）から始まる文字列の符号化を開始する。

【００４７】登録制御において一致長を基準にする場合、一致長が閾値よりも大きかったら登録を控えるのが妥当である。閾値は経験的には４から１６程度が良い結果をもたらす。動的に閾値を変化させる場合は、一致長のそれまでの平均値を閾値とする方法が考えられる。図３は平均値を用いて閾値を動的に変化させる場合の登録制御手段の処理の流れを示すフローチャートである。

【００４８】図４は従来方式と一致長を基準にする登録制御方式を比較した図である。図２の処理の流れに沿って、図４の状況での一致長を基準とした登録制御の具体例を説明する。履歴アレイ手段２２は１２個のエントリから成っている。履歴アレイ手段２２の中で灰色の部分

は先読み領域で、これから符号化を行う文字列を格納しており、今は履歴アレイ手段の８から１１の位置のエントリである。履歴アレイ手段２２の先読み領域以外の位置から始まる文字列はすべて検索手段に登録されているとして、インデックスアレイ手段２１は‘０’から順に‘７’までのインデックスを格納しているとする。先読み領域の長さが４のため、文字列の検索手段への登録は、ある位置から始まる長さ４の文字列を登録すると、という形で行う。このとき、履歴アレイ手段２２内で先読み領域の先頭から始まる文字列（圧縮を行おうとしている文字列）と検索手段に登録されている文字列との間で最長一致列を求めると、“ｂｃｂ”という文字列が得られる（ステップＳ３）。このとき８から始まる文字列は、一致列を求める過程で検索手段に登録される。インデックスアレイ手段２１の対応するエントリには新しいインデックスである‘８’が格納される（ステップＳ４）。“ｂｃｂ”に対する符号語は一致長‘３’（先読み領域の先頭のインデックスと比較して３個前のインデックスで示される位置、という情報）を元に生成される（ステップＳ６）。符号語の出力が済んだら、履歴アレイ手段２２を３文字分更新する。先読み領域は１１から始まる領域に移動する。履歴アレイ手段を円環的に使用していたため、先読み領域は（１１，０，１，２）の位置のエントリに移動する。０，１，２から始まる文字列は検索手段から削除される。その位置のインデックスも無効になる。履歴アレイ手段２２の（０，１，２）の位置には新たに読み込んだデータである“ｃａａ”が挿入される（ステップＳ９）。図４（ａ）は登録制御が行われない場合、図４（ｂ）は登録制御が行われる場合の状況を表している。両者とも履歴アレイ手段の状態は同一で、２４または２６で表される。登録制御では、履歴アレイ手段２４（または２６）の９，１０から始まる文字列（“ｃｂｂｃ”，“ｂｂｃａ”）を検索手段に登録するかどうかが問題になる（ステップＳ９）。従来方式は９，１０から始まる文字列を検索手段に登録する（図４（ａ））。対応するインデックス‘９’，‘１０’がインデックスアレイ手段２３に格納される。一致長を基準にした登録制御では、一致長が３のため、閾値が４以上であったら従来方式と同様に検索手段への登録、インデックスアレイ手段２３の更新を行い（図４（ａ），ステップＳ１０）、閾値が４よりも小であったら９，１０から始まる文字列の登録は行わない（図４（ｂ））。インデックスアレイ手段２５の９，１０のエントリは空になる。図４（ａ）の状況で、次に１１の位置に格納されるインデックスは‘１１’であるが、図４（ｂ）では‘９’となる。

【００４９】登録制御において一致列の開始点を基準にする場合、一致列の開始点の位置が閾値よりも小さかったら登録を控えるのが妥当である。動的に閾値を変化させる方法は一致長の場合と同様に平均値を使用する方法が考えられる。

【００５０】図５は従来方式と一致位置を基準にする登録制御方式を比較した図である。図５の状況での一致位置を基準とした登録制御の具体例を説明する。図５の履歴アレイ手段３２やインデックスアレイ手段３１の状況は図４と同様である。また、最長一致列を求め、履歴アレイ手段を更新する状況も図４と同様である。図５（ａ）は登録制御が行われない場合、図５（ｂ）は登録制御が行われる場合の状況を表している。両者とも履歴アレイ手段の状態は同一で、３４または３６で表される。登録制御では、履歴アレイ手段３４（または３６）の９，１０から始まる文字列（“ｃｂｂｃ”，“ｂｂｃａ”）を検索手段に登録するかどうかが問題となる。従来方式は図４の場合と同様に履歴アレイ手段３４の９，１０から始まる文字列を検索手段に登録する（図５（ａ））。対応するインデックス‘９’，‘１０’がインデックスアレイ手段３３に格納される。一致位置を基準にした登録制御では、一致位置が先読み領域の先頭からみて３個前の位置のため、閾値が４以上であったら従来方式と同様に検索手段への登録、インデックスアレイ手段３３の更新を行い（図５（ａ）），閾値が４よりも小であったら９，１０から始まる文字列の登録は行わない（図５（ｂ））。インデックスアレイ手段３５の９，１０のエントリは空になる。図５（ａ）の状況で、次に１１の位置に格納されるインデックスは‘１１’であるが、図５（ｂ）では‘９’となる。

【００５１】次に、本発明の登録制御手段のもう一つの実施例について説明する。図６はこの実施例の登録制御の流れを示す図である。先頭ｘ（ｓ）から始まる文字列は先に述べたようにこの符号化のための検索結果を用いて既に登録されているとする。今一致列がｘ（ｓ）…ｘ（ｔ）（一致長Ｌ＝ｔ－ｓ＋１）であったとする（ステップＴ１）。リテラルモードで符号化された場合（Ｌ＝１）、または閾値ｒ＝１の場合には先頭のｘ（ｓ）から始まる文字列のみが検索手段２に登録されるので、一致列の途中から始まる文字列の登録はない（ステップＴ２）。そうでない場合、一致列として切り出された文字列の中で、最初のｒ個の位置から始まる文字列のみを検索手段２に登録する。つまり、ｘ（ｓ）…ｘ（ｔ）（Ｌ＝ｔ－ｓ＋１）の途中から始まる文字列としては、ｘ（ｓ＋１），…，ｘ（ｓ＋ｒ－１）から始まる文字列のみを検索手段２に登録するような登録制御を行うのである（ステップＴ４）。ただし、Ｌ＜ｒの場合にはｘ（ｓ＋１），…，ｘ（ｓ＋Ｌ－１）（＝ｘ（ｔ））から始まる文字列を検索手段に登録する（ステップＴ３，Ｔ５）。閾値ｒの設定を変更することで符号化速度と圧縮率のどちらを重視するかを調節するようなアプリケーションも考えられる。この場合、圧縮データの先頭にこの閾値を記しておくことにより、復元側ではこの閾値を知ることができ、この値さえ知っていれば、履歴アレイ中のどの

13

位置から始まる文字列が登録されているのかを正しく知ることができるため正しく復元できる。

【００５２】図７は従来方式と一致列の先頭から閾値個までの文字列を登録する方式を比較した図である。図７の状況での本登録制御方式の具体例を説明する。図７の履歴アレイ手段４２やインデックスアレイ手段４１などの状況は図４と同様である。また、最長一致列を求め、履歴アレイ手段を更新する状況も図４と同様である。図７（ａ）は登録制御が行われない場合、図７（ｂ）は登録制御が行われる場合の状況を表している。両者とも履歴アレイ手段の状態は同一で、４４（または４６）で表される。登録制御では、履歴アレイ手段４４（または４６）の９，１０から始まる文字列（“ｃｂｃ”，“ｂｃａ”）を検索手段に登録するかどうかが問題になる。従来方式は履歴アレイ手段４４の９，１０から始まる文字列を検索手段に登録する（図７（ａ））。対応するインデックス‘９’，‘１０’がインデックスアレイ手段４３に格納される。閾値を２とした場合、先頭の次の位置である、履歴アレイ手段４６の９の位置から始まる文字列が検索手段に登録される（図７（ｂ））。インデックスアレイ手段４６の９の位置のエントリは‘９’が格納され、１０の位置のエントリは空となる。図７（ａ）の状況で、次に１１の位置に格納されるインデックスは‘１１’であるが、図７（ｂ）では‘１０’となる。

【００５３】また、更に閾値ｒが２以上のときには次のように圧縮率の改善をみることができる。ｒ≧２の時には符号化する文字列の先頭ｘ（ｓ）の次の文字ｘ（ｓ＋１）から始まる文字列を検索手段２に登録する必要があるため、木構造などを検索手段に用いた場合にはその過程でｘ（ｓ＋１）から始まる文字列に対する一致列を求める必要がある。そこで、ｘ（ｓ）からの一致列とｘ（ｓ＋１）からの一致列を比較してｘ（ｓ＋１）からの一致列の方が大きかったらｘ（ｓ）をリテラルモードで符号化し、ｘ（ｓ＋１）からの一致列をコピーモードで符号化する。こうすることによって、２回のコピーモードで符号化するところを１回のリテラルモードと１回のコピーモードで符号化できる場合が多い。リテラルモードの方がコピーモードよりも少ないビット数で表現できるため圧縮率が向上するのである。この処理の詳細な説明を図８に示す。

【００５４】ｘ（ｓ）からの一致列がｘ（ｓ）…ｚ（ｓ＋ｌ－１）（ｌは一致長）であったとする（ステップＵ１）。ｌが十分に大きくないときにはｘ（ｓ）をリテラルモードで符号化し（ステップＵ３）。履歴アレイ手段１の更新、履歴アレイ手段１から溢れた文字列を検索手段２から溢れた文字列の削除を行い（ステップＵ８）、ｘ（ｓ＋１）の符号化に移る（ステップＵ１３）。ｌが十分に大きいときには、先頭の次の文字ｘ（ｓ＋１）から始まる文字列と履歴アレイ手段１内の文字列との最長

14

一致列を求め、その検索結果を用いてｘ（ｓ＋１）から始まる文字列を検索手段２に登録する（ステップＵ４）。一致列をｘ（ｓ＋１）…ｘ（ｓ＋ｌ′）（ｌ′はｌ一致長とする）。ｌとｌ′を比較し（ステップＵ５）、ｌ′の方が大きかったらｘ（ｓ）をリテラルモードで、ｘ（ｓ＋１）…ｘ（ｓ＋ｌ′）をコピーモードで符号化する（ステップＵ９）。符号化された分の大きさ（（ｌ＋ｌ′）個の文字分）履歴アレイ手段１を更新し、履歴アレイ手段１から溢れた文字列の削除を行う（ステップＵ９）。ｘ（ｓ＋１）…ｘ（ｓ＋ｌ′）から始まる文字列に対して図６に示した登録制御法で検索手段２への登録を行う（ステップＵ１１）。そして、ｘ（ｓ＋ｌ′＋１）から始まる文字列の符号化に移る（ステップＵ１４）。ｌがｌ′以上の場合にはｘ（ｓ）…ｘ（ｓ＋ｌ－１）をコピーモードで符号化する（ステップＵ７）。符号化された分の大きさ（ｌ個の文字分）履歴アレイ手段を更新し、履歴アレイ手段１から溢れた文字列の削除を行う（ステップＵ１０）。ｘ（ｓ）…ｘ（ｓ＋ｌ－１）から始まる文字列に対して図６に示した登録制御法で検索手段２への登録を行う（ステップＵ１２）。このとき、ｘ（ｓ＋１）から始まる文字列の登録は既に済んでいることに注意。そして、ｘ（ｓ＋１）から始まる文字列の符号化に移る（ステップＵ１５）。

【００５５】以上のように登録制御を行うと、検索手段に登録されている履歴アレイ手段内の文字列は連続して現れることが多くなる。ここで“連続して現れる文字列”は、隣合う位置から始まる文字列、ということを意味する。次に、このことを利用して、圧縮率を向上させる方式を図９を用いて説明する。

【００５６】今、履歴アレイ手段内でｘ（ｐ），ｘ（ｐ＋１），…ｘ（ｒ）から始まる文字列は検索手段に登録されており、ｘ（ｐ－１），ｘ（ｒ＋１）から始まる文字列は検索手段に登録されていないとする。このとき、ｘ（ｐ）…ｘ（ｒ）を“ブロック”と呼ぶことにする。図９に示すように、履歴アレイ手段５２の先読み領域以外のエントリは、いくつかのブロックと検索手段に登録されていない部分に分けられることになる。このとき現れたブロックには順にインデックスを割り振り、インデックスアレイ手段５１はこのブロックのインデックスを格納する。図９に示したように、同じブロック内のエントリは同一のインデックスをインデックスアレイ手段５１に格納している。従来方式では履歴アレイ手段内の先読み領域以外の位置から始まるすべての文字列が検索手段に格納されているため、先読み領域以外の部分は一つの大きなブロックになる。

【００５７】先読み領域内の文字列をｘ（ｓ）ｘ（ｓ＋１）…とし、ｘ（ｓ）ｘ（ｓ＋１）…と履歴アレイ手段内で検索手段に登録されている文字列との最長一致列がｘ（ｓ－ｋ）…ｘ（ｓ－ｋ＋ｌ－１）（＝ｘ（ｓ）…ｘ（ｓ＋ｌ－１））であったとする。前述の方式では一致

位置の情報をｘ（ｓ－ｋ）に対応するインデックスに基づいて生成したが、このブロックを用いた方式では、ｘ（ｓ－ｋ）の属するブロックのインデックスとｘ（ｓ－ｋ）のそのブロック内での位置の情報に基づいて生成する。具体的には、ｘ（ｓ－ｋ）が先読み領域からみてｒ個前のブロックに属し、そのブロックの最後のエントリにはｘ（ｓ－ｋ＋ｕ）が格納されているとすると、（ｒ，ｕ）という情報でｘ（ｓ－ｋ）の位置を指定する。図９は一致長が５で、ｒ＝３，ｕ＝０である状況を表している。ｒはインデックスアレイ手段５１から求められる。ｕはブロックのインデックスに対してそのブロックの最後のエントリの位置を格納するアレイを用意する方法か、またはインデックスアレイ手段５１を用いて同じインデックスが続くところまで検索する方法によって求めることができる。

【００５８】一般に複数のエントリが集まってブロックを形成するため、ブロックの数は検索手段に登録されているエントリの数に比べてはるかに小さいものとなる。ブロックのインデックス情報（一致列はｒ個前のブロックから始まる、というｒの値）の分布は、個々のエントリに割り振ったインデックス情報（一致列はｒ′個前のインデックスの位置から始まる、というｒ′の値）の分布に比べて小さい値に集中するため、ブロックのインデックスの符号長の方が小さい値で表せることになる。一方、ブロックのインデックスを用いる方法はブロック内の位置を表す情報も付加しなければならないため、全体としてみれば個々のエントリにインデックスを割り振る方法と圧縮率は変わらない場合もある。しかし、例えば次に示すような場合に圧縮率の改善をみることができる。

【００５９】データ中にランダムに近い大きな領域があると、リテラルモードの出力が続く。つまりその部分の文字列は連続して検索手段に登録され、大きなブロックを形成する。しかし、そのブロックから始まる文字列はその後参照される可能性は低い。各エントリにインデックスを割り振った場合には、そのような大きなブロックより遠い位置に割り振られたインデックスは、最新のインデックスと大きく離れた値となる。しかし、ブロックのインデックスは最新のブロックのインデックスとそれほど離れた値とはならない。ブロックのインデックスを用いて一致位置を表す方法は、このように大きな無駄な領域を飛び越えて一致位置を効果的に表す方法となるのである。

【００６０】検索手段として二分木を用い、履歴アレイ手段の大きさＮ＝１６３８４、一致列の最大長Ｌ＝２５６として２メガバイトのＣプログラムソースを圧縮する実験を行った。従来の方法では１７５秒の実行時間で２８．１％の圧縮率を達成した。図３に示した登録制御を行った本発明では、３２秒の実行時間で２９．２％の圧縮率を達成した。つまり、１％未満の圧縮率の劣化で６倍の符号化速度の向上を得ている。また、図８に示した本発明において閾値を２に設定して登録制御処理を行った本発明の方式では３２秒の実行時間で２８．７％の圧縮率を達成し、図３の登録制御法と同じ圧縮時間で０．５％程度優れた圧縮率を達成している。また、Ｎ＝８１９２，Ｌ＝３２と変更して従来の方法と同じデータを圧縮したときには、６７秒の実行時間で３１．０％の圧縮率であった。つまり、検索手段として二分木を用い、高速な符号化を求めるのなら、Ｎ，Ｌを変更するよりも、本発明の方式を採用した方が有効であり、かつ圧縮率の劣化も小さいことがわかる。

【００６１】また、図９に示したブロックのインデックスを用いた符号化法で等長符号化を行った場合の効果は次のように評価できる。検索手段に登録されているエントリの数をＭ、ブロックの平均長をＢとする。ブロックの数はＭ／Ｂとなる。個々のエントリにインデックスを順に割り振った場合、インデックスを表すためのビット数はほぼｌｏｇＭビットとなる。一方、ブロックのインデックスを表すためのビット数はほぼｌｏｇ（Ｍ／Ｂ）＝ｌｏｇＭ－ｌｏｇＢビットとなる。ここでｌｏｇの底は２である。ブロックがどれも同じ大きさであったら、ブロック内での位置を表すためにほぼｌｏｇＢビット必要になり、この場合はブロックのインデックスを用いた方式を用いても圧縮率は変わらない。しかし、履歴アレイ手段中にランダムに近い大きな領域がある場合、その領域からなるブロックを除いた、実際に後に参照される文字列を含むブロックの平均長Ｂ′はＢよりも小さくなる。そこで、ブロックのインデックスを用いる方式によって一回の一致位置情報の出力につき（ｌｏｇＢ－ｌｏｇＢ′）ビットだけ圧縮データを短縮化できるのである。

【００６２】以上本発明のデータ圧縮方式について説明したが、本発明は、データ復元方式にも応用できる。データ復元方式の実施態様を以下に説明する。

（１）既に復元を終えたデータを格納する複数のエントリを有する履歴アレイ手段を有し、前記履歴アレイ手段における位置、長さを表す圧縮符号語に対応する文字列を複製することで元データを復元するデータ復元方式において、各エントリが前記履歴アレイ手段の各エントリに対応したインデックスアレイ手段を有し、圧縮データに対応する可能性がある文字列は前記履歴アレイ手段の制限された位置から始まる文字列のみであり、前記インデックスアレイ手段は、圧縮データに対応する可能性がある文字列の先頭位置にのみ割り振られたインデックスを保持し、圧縮データにおける前記インデックスから生成された圧縮符号語から複製される文字列の開始点を確定することを特徴とするデータ復元方式。

（２）圧縮データに対応する可能性がある前記履歴アレイ手段内の文字列の先頭位置は、前記復元文字列の長さに基づいて決定されることを特徴とする（１）記載のデ

ータ復元方式。

（３）圧縮データに対応する可能性がある前記履歴アレイ手段内の先頭位置は、復元された文字列の先頭位置と前記復元文字列の長さが予め決められた閾値以下であった場合の前記復元文字列の途中の位置であることを特徴とする（２）記載のデータ復元方式。

（４）前記閾値を動的に変更することを特徴とする（３）記載のデータ復元方式。

（５）圧縮データに対応する可能性がある文字列の先頭位置は、圧縮データで指示される前記履歴アレイ手段内の復元文字列と同一の文字列の先頭の前記アレイ手段に基づいて決定されることを特徴とする（１）記載のデータ復元方式。

（６）圧縮データに対応する可能性がある前記履歴アレイ手段内の文字列の先頭位置は、復元文字列の先頭位置と、圧縮データによって指示される復元文字列と同一の文字列の先頭の前記履歴アレイ手段における位置と、前記復元文字列の先頭の位置との相対距離が予め決められた閾値以上であった場合の前記復元文字列の途中の位置であることを特徴とする（５）記載のデータ復元方式。

（７）閾値を動的に変更することを特徴とする（６）記載のデータ復元方式。

（８）圧縮データに対応する可能性がある前記履歴アレイ手段内の先頭位置は、過去に復元された文字列の中で先頭位置から閾値以下の順位の位置に制限されていることを特徴とする（１）記載のデータ復元方式。

（９）前記履歴アレイ手段中の圧縮データに対応する可能性がある文字列うち、連続している文字列の先頭エントリをブロックとしてまとめ、同じ該ブロックに属するエントリには同じインデックスを割り振り、前記インデックスアレイ手段はこれを格納し、圧縮符号語から該ブロックのインデックスと該ブロック内での位置の情報が求められ、一致列の開始位置が定まることを特徴とする（１）記載のデータ復元方式。

【００６３】

【発明の効果】以上説明したように本発明のデータ圧縮方式を用いれば、大きな履歴アレイを用いても比較すべき文字列の個数、検索手段へ登録する文字列の数が削減でき、符号化速度の向上を図ることができる。特に冗長の大きなデータに対しては大きな符号化速度の向上を生む。更に、登録制御を行うことを積極的に利用して、検索手段に登録されている文字列の先頭エントリ、または隣接する先頭エントリをまとめたブロックに割り振られたインデックスを用いて一致位置情報を表すことにより、圧縮率の劣化を小さく抑えることができる。このように本発明によって圧縮率を劣化させずに符号化速度の向上を達成できるのである。

【００６４】また、登録に検索と同じ時間が必要となる木構造を用いた検索手段に関しては本発明のように選択的に検索構造へ登録することは極めて効果がある。ハッシュテーブルを用いた方法でも、ハッシュ関数を計算する回数が少なくなるとともに、検索すべきアレイ内の文字列の数も減少し、線形リストを辿る回数が減少するために効果がある。このように、本発明の登録制御は様々な検索手段に対応させることができる。

【図面の簡単な説明】

【図１】本発明の一実施例を示すブロック図である。

【図２】本発明の処理の流れを示す図である。

【図３】動的に閾値を変更する処理の流れを示す図である。

【図４】一致長を基準にして登録制御を行う方式の具体例を、従来方式と比較して示した図である。

【図５】一致位置を基準にして登録制御を行う方式の具体例を、従来方式と比較して示した図である。

【図６】閾値個以下の位置から始まる文字列のみを検索手段に登録する場合の登録制御手段の処理の流れを示す図である。

【図７】閾値個以下の位置から始まる文字列のみを検索手段に登録する方式の具体例を、従来方式と比較して示した図である。

【図８】閾値個以下の位置から始まる文字列のみを検索手段に登録する方式で、閾値が２以上の場合の改良型の登録制御手段および符号化データ生成手段の処理の流れを示す図である。

【図９】ブロックのインデックスによって一致位置を表す方式を示す図である。

【図１０】従来の圧縮方式の構成を示す図である。

【図１１】履歴アレイ手段を示す図である。

【図１２】二分木を用いた方法を示す図である。

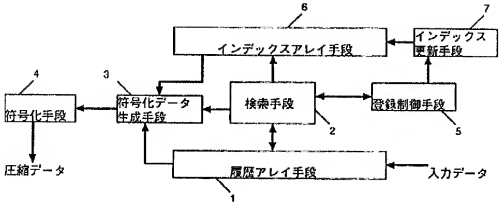
【図１３】トライを用いた方法を示す図である。

【図１４】ハッシュテーブルに衝突用の線形リストを備えた方法を示す図である。

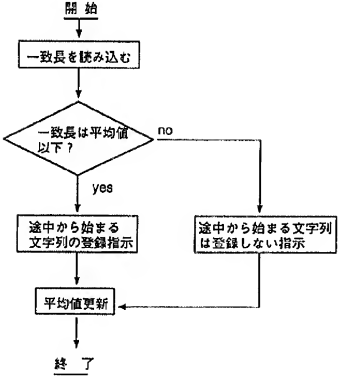
【符号の説明】

１　履歴アレイ手段
２　検索手段
３　符号化データ生成手段
４　符号化手段
５　登録制御手段
６　インデックスアレイ手段
７　インデックス更新手段

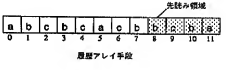
【図１】



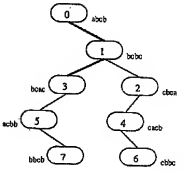
【図３】



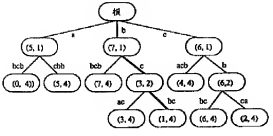
【図１１】



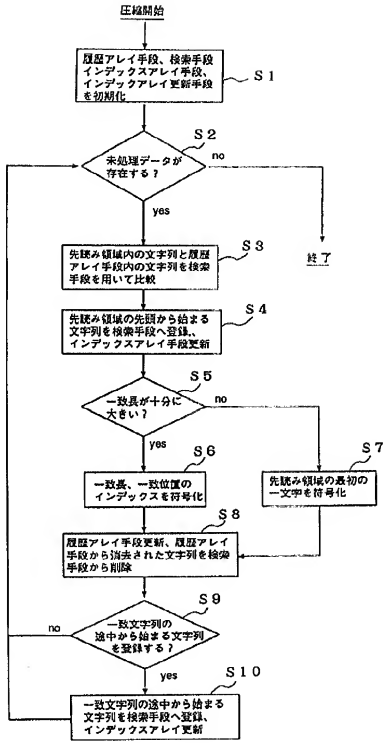
【図１２】



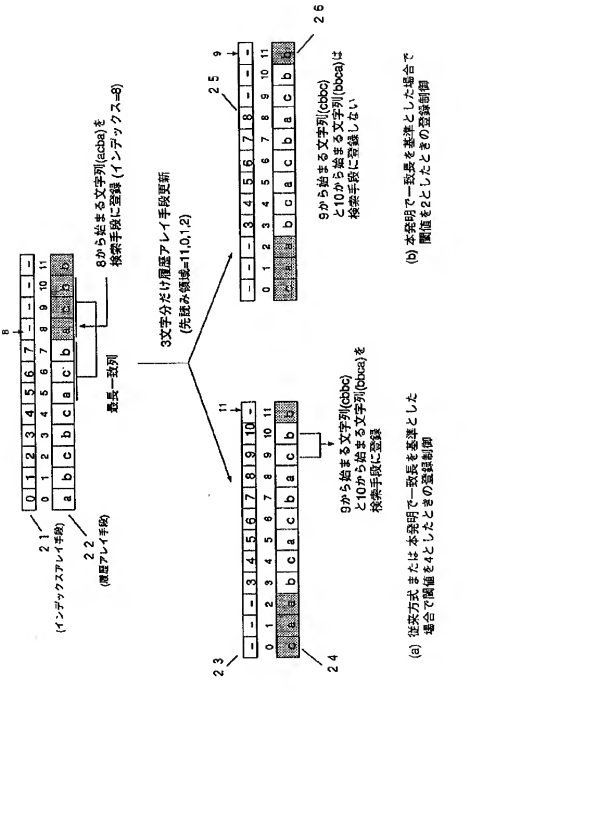
【図１３】



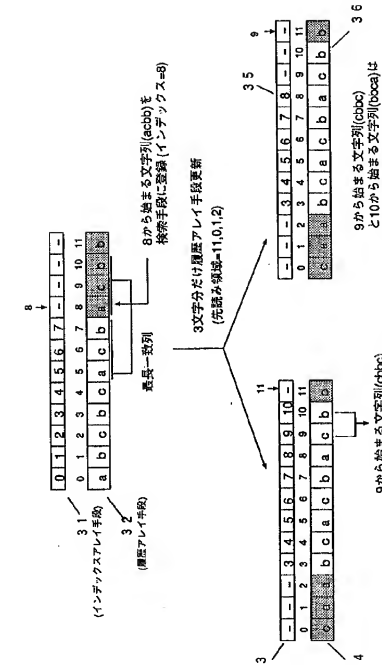
【図２】


圧縮開始
履歴アレイ手段、検索手段
インデックスアレイ手段、
インデックアレイ更新手段
を初期化
S１
S２
未処理データが
存在する？
no
yes
終了
先読み領域内の文字列と履歴
アレイ手段内の文字列を検索
手段を用いて比較
S３
先読み領域の先頭から始まる
文字列を検索手段へ登録、
インデックスアレイ手段更新
S４
S５
一致長が十分に
大きい？
no
yes
S６
一致長、一致位置の
インデックスを符号化
S７
先読み領域の最初の
一文字を符号化
S８
履歴アレイ手段更新、履歴アレイ
手段から消去された文字列を検索
手段から削除
S９
一致文字列の
途中から始まる文字列
を登録する？
no
yes
S１０
一致文字列の途中から始まる
文字列を検索手段へ登録、
インデックスアレイ更新

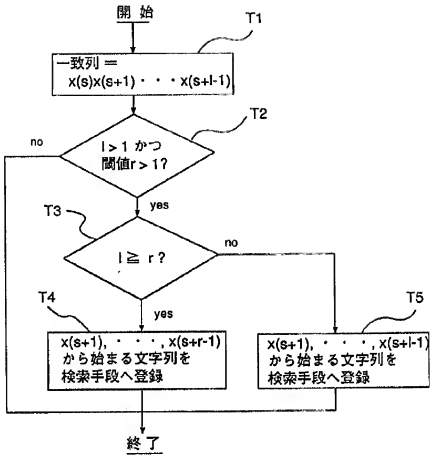
【図4】


(インデックスアレイ手段)
(履歴アレイ手段)
8から始まる文字列(acbaa)を
検索手段に登録(インデックス=8)
最長一致列
3文字分だけ履歴アレイ手段更新
(先読み領域=11,0,12)
9から始まる文字列(cbbc)
と10から始まる文字列(bbca)を
検索手段に登録
9から始まる文字列(cbbc)
と10から始まる文字列(bbca)は
検索手段に登録しない
(a) 従来方式 または本発明で一致長を基準とした
場合で閾値を4としたときの登録前例
(b) 本発明で一致長を基準とした場合で
閾値をとしたときの登録前例

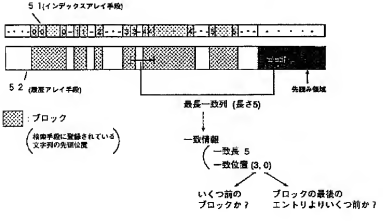
【図5】


(インデックスアレイ手段)
31
(履歴アレイ手段)
32
最長一致列
8から始まる文字列(acbb)を
検索手段に登録(インデックス=8)
3文字分だけ履歴アレイ手段更新
(先読み領域=11,0,1,2)
33
34
9から始まる文字列(cbbc)
と10から始まる文字列(bcca)を
検索手段に登録
35
36
9から始まる文字列(cbbc)
と10から始まる文字列(bcca)は
検索手段に登録しない
(a) 従来方式または本発明で一致位置を基準とした
場合で閾値を4としたときの登録制御
(b) 本発明で一致位置を基準とした場合で
閾値を2としたときの登録制御

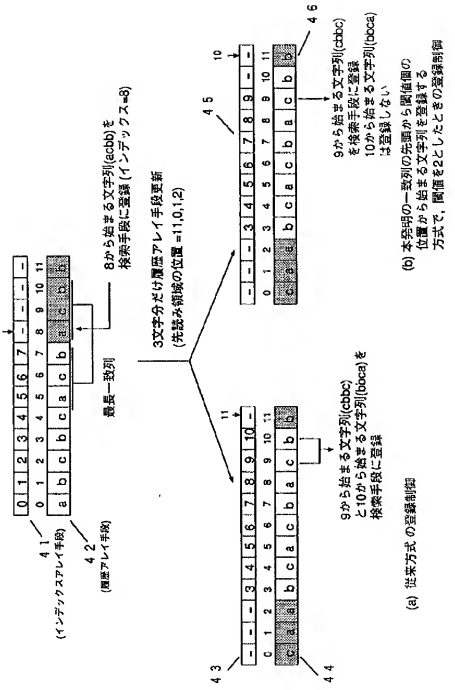
【図６】


開 始
T1
一致列＝
x(s)x(s+1)・・・x(s+l-1)
T2
no
l＞1 かつ
閾値r＞1？
yes
T3
l≧ r？
no
T4
yes
T5
x(s+1), ・ ・ ・, x(s+r-1)
から始まる文字列を
検索手段へ登録
x(s+1), ・ ・ ・, x(s+l-1)
から始まる文字列を
検索手段へ登録
終了

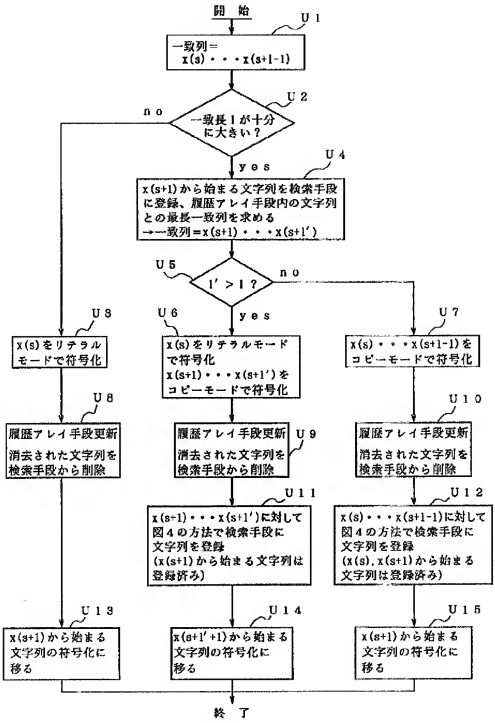
【図９】


５１(インデックスアレイ手段)
５２
(提示アレイ手段)
先読み領域
最長一致列（長さ5）
一致情報
一致長 5
一致位置 (3, 0)
：ブロック
検索手段に登録されている
文字列の先頭位置
いくつ前の
ブロックか？
ブロックの最後の
エントリよりいくつ前か？

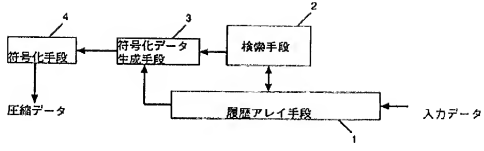
【図7】


8
0 1 2 3 4 5 6 7 - - - -
0 1 2 3 4 5 6 7 8 9 10 11
a b c b c a c b a c b b
41
(インデックスアレイ手段)
42
(履歴アレイ手段)
最長一致列
8から始まる文字列(acbb)を
検索手段に登録(インデックス=8)
3文字分だけ履歴アレイ手段更新
(先読み領域の位置 =11,0,12)
43
11
9から始まる文字列(cbbc)
と10から始まる文字列(bcca)を
検索手段に登録
44
45
10
9から始まる文字列(cbbc)
を検索手段に登録
10から始まる文字列(bcca)
は登録しない
46
(a) 従来方式の登録制御
(b) 本発明の一致列の先頭から閾値個の
位置から始まる文字列を登録する
方式で，閾値をとしたときの登録制御

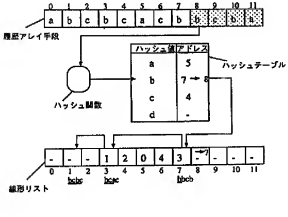
【図8】


開　始
U1
一致列＝
x(s)・・・x(s+l-1)
U2
一致長 l が十分に大きい？
no
yes
U4
x(s+1)から始まる文字列を検索手段に登録、履歴アレイ手段内の文字列との最長一致列を求める
→一致列＝x(s+1)・・・x(s+l')
U5
l' > l ?
no
yes
U3
x(s)をリテラルモードで符号化
U6
x(s)をリテラルモードで符号化
x(s+1)・・・x(s+l')をコピーモードで符号化
U7
x(s)・・・x(s+l-1)をコピーモードで符号化
U8
履歴アレイ手段更新
消去された文字列を検索手段から削除
U9
履歴アレイ手段更新
消去された文字列を検索手段から削除
U10
履歴アレイ手段更新
消去された文字列を検索手段から削除
U11
x(s+1)・・・x(s+l')に対して図4の方法で検索手段に文字列を登録
(x(s+1)から始まる文字列は登録済み)
U12
x(s)・・・x(s+l-1)に対して図4の方法で検索手段に文字列を登録
(x(s),x(s+1)から始まる文字列は登録済み)
U13
x(s+1)から始まる文字列の符号化に移る
U14
x(s+l'+1)から始まる文字列の符号化に移る
U15
x(s+1)から始まる文字列の符号化に移る
終　了

【図１０】


4
3
2
符号化手段
符号化データ
生成手段
検索手段
圧縮データ
履歴アレイ手段
入力データ
1

【図１４】


0 1 2 3 4 5 6 7 8 9 10 11
a b c b c a c b
履歴アレイ手段
ハッシュ値アドレス
a 5
b 7
c 4
d -
ハッシュテーブル
ハッシュ関数
- - - 1 2 0 4 3 - - - -
連結リスト
hdc
hcc
hbcb